ISSN 2007-9737

An Approach for Prototype Generation based on Similarity Relations for Problems of Classification

Yumilka B. Fernández Hernández[1], Rafael Bello[2], Yaima Filiberto[1], Mabel Frías[1], Lenniet Coello Blanco[1], and Yaile Caballero[1]

[1] Departamento de Ciencias de la Computación, Universidad de Camagüey, Camagüey, Cuba

[2] Departamento de Ciencias de la Computación, Universidad Central de Las Villas, Cuba

{yumilka.fernandez, yaima.filiberto, lenniet.coello, mabel.frias, yaile.caballero}@reduc.edu.cu, rbellop@uclv.edu.cu

Abstract. In this paper, a new method for solving classification problems based on prototypes is proposed. When using similarity relations for granulation of a universe, similarity classes are generated, and a prototype is constructed for each similarity class. Experimental results show that the proposed method has higher classification accuracy and a satisfactory reduction coefficient compared to other well-known methods, proving to be statistically superior in terms of classification's precision.

Keywords. Prototype generation, similarity relations, classification.

1 Introduction

Learning methods for problems of instance-based classification use a training set to estimate the class label(s). These learning algorithms have scalability problems when the training set size grows, so the number of training instances affects the computational cost of a method [1]; as shown in [2] the nearest neighbor rule is an example of a high computational cost method when the number of instances is big.

An alternative to mitigate this problem is the classification based on the Nearest Prototype (NP) [3]. This is a method to determine the value of the decision attribute of a new object by analyzing its similarity with respect to a set of prototypes selected or generated from the initial set of instances.

The purpose of the NP approach is to reduce storage costs and learning technique processes based on instances. In [4] it is stated that when the amount of information is large, a possible solution is to reduce the number of "example" vectors provided that the efficiency obtained is better or the same as when the original data set is used. So a good set of prototypes has two important characteristics: a minimal cardinality and a high accuracy in solving a problem. Strategies are needed to reduce the number of examples of the input data to a small representative number; its performance should be assessed taking into account the classification accuracy and reduction coefficient.

It can be said that the methods of data reduction with respect to instances are divided into two categories: Prototype Selection (PS) [5] and Prototype Generation (PG) [6]. The prototype selection algorithms choose a set of representative instances according to a selection criterion, and they can improve the predictive power of the classifier according to the nearest neighbor rule [7, 8, 9], while the algorithms for prototype generation are able to generate a set of new objects of the application domain from the initial instances.

The way to get this set of prototypes is based on selecting an original set of labeled examples or replacing the original set by a different and diminished one [4, 6].

The prototype generation aims at obtaining a training set as reduced as possible for classifying

with the same or higher quality as that of the original training set. This reduces the space complexity of the method and also its computational cost. Sometimes accuracy can be also improved by eliminating noise. The prototype generation techniques have proven to be very competitive by improving the performance of the nearest neighbor classifier [10]. Prototype-based classifiers allow determining the class of a new example based on a reduced prototype set instead of using a large set of known examples.

Concerning related work, [11] proposed the NPBASIR method for functions; since this method showed good results, we used it as a basis for developing our NP-BASIR-Class algorithm for classification problems.

So this paper proposes the NP-BASIR-Class method for constructing prototypes for classification problems using the concepts of Granular Computing [12] based on NP-BASIR [11].

Granulation of a universe is performed using a relation of similarity which generates similarity classes of objects in the universe, and for each similarity class one prototype is built. To construct the similarity relation, the method proposed in [13] is used. In Section 2, our method of prototype construction and classifier are described; in Section 3, we compare the performance of our classifier with other prototype-based classifiers, and at the end of the paper we present conclusions.

2 Related Work

In the literature, there are different strategies to find a good set of prototypes. In [14], a hybrid approach is proposed to integrate a weighting scheme in order to obtain data reduction, with a self-adaptive differential evolution technique for optimizing the weighting given to each feature and the location of the prototypes, obtaining improvements on the performance of the nearest neighbor classifier.

Several approaches are based on clustering techniques [15, 18] which are characterized by two main stages: the first stage is to group a set of input data without labels and obtain a reduced set of prototypes; the second one is to add labels to these prototypes based on the already labeled examples and the NP rule [19].

In [20], an evolutionary approach based on the Nearest Prototype classifiers is proposed; the algorithm is based on the evolution of a set of prototypes that can execute several operators to enhance quality in a local sense and a high classification accuracy that arises for the whole classifier. A work with a radial basis function is reported in [21]. Genetic algorithm approaches are commonly used to find an initial set of prototypes and its correct size.

In [22] a new method is proposed for finding prototypes based on clustering; it is more efficient in large databases, where the main objective is to divide the data set into regions using clustering, analyze prototypes belonging to different classes and when a particular prototype belongs to a single class, preserve this prototype.

In [6], an experimental study is performed in which the main features of the prototype generators are identified; their taxonomy is also displayed in a hierarchical order based on the generation mechanisms, the resulting generation sets, and the evaluation search.

3 Method for the Construction of Prototypes

The proposed method is an iterative procedure in which prototypes are constructed from similarity classes of objects in the universe: a similarity class is constructed using the similarity relation R $([O_i]R)$ and a prototype is constructed for this class of similarity. Whenever an object is included in a class of similarity, it is marked as used, and is not taken into account when another similarity class is constructed; but used objects can belong to similarity classes that will be constructed for other non-used objects. An arrangement of n components is used, called Used[n] wherein; Used[i] has a value of 1 if the object was already used and 0 otherwise.

The proposed method uses a similarity relation R and a set of instances $X = \{X_1, X_2 \ldots X_n\}$, each of which is described by a vector of m descriptive features and belongs to one of k classes $C = \{c_1, c_2, \ldots, c_k\}$. The similarity relation R is constructed

according to the method proposed in [13]; this is based on finding the relation that maximizes the quality of the similarity measure. In this case, the relation *R* is sought that generates a granulation considering the *m* descriptive features, as similar as possible to the granulation according to the classes.

Algorithm: NP-BASIRC (input: *X*, output: *ProtoSet*)

Given a set of *n* objects with *m* descriptive features and one decision feature, and a similarity relation *R*:

P1: *Initialize objects' counter.*
Used [j] ← 0, for j=1,..., n
ProtoSet ← ∅, i ← 0
P2: *Start processing the object O_i.*
i ← index of the first non-used object
If i=0 then End of the generation of prototypes.
P3: *Construct the similarity class of object O_i according to R.*
Construct $[O_i]R$
[x]R denotes the similarity class of object x
P4: *Construct a vector P with m components (one for each descriptive feature) for all objects in $[O_i]R$.*
P(i) ← f(V_i), where V_i is the set of values of feature i in objects in $[O_i]R$ and f is an aggregation operator.
ProtoSet ← ProtoSet ∪ P
P5: *Mark as used all objects in the similarity class $[O_i]R$.*
Used [j] ← 1 for all $O_j \in [O_i]R$
P6: *Go to P2*

In step P4, the function *f* denotes an aggregation operator, for example, if the values in V_i are real, the average may be used; if they are discreet, the most common value is used. The purpose is to build a prototype or centroid for a set of similar objects.

The set of prototypes *ProtoSet* is the output of the NP-BASIRC algorithm. This set is used by the classifier to classify new instances:

Algorithm NP-BASIR-Class (input: *ProtoSet*, output: class)

Given a new object x and the set ProtoSet:

P1: *Calculate the similarity [22] between x and each prototype.*
P2: *Select the k most similar prototypes.*
P3: *Calculate the class of x as the most common class in the set of k most similar prototypes.*

In step 3, the class is calculated in the same way as k-nearest neighbors (k-NN) for classification are calculated [23].

4 Experimental Results

The performance of the proposed method in the previous section has been studied using the relation *R* defined by expression (1), that is, two objects are similar if their similarity according to the descriptive features is greater than a threshold ε and they belong to the same class:

$$xRy \Leftrightarrow F1(x, y)) \geq \varepsilon \text{ and } F2(x, y) = 1 \quad (1)$$

where functions F1 and F2 are defined by expressions (2) and (3), respectively:

$$F1(x, y) = \sum_{i=1}^{n} w_i \times \partial_i (x_i, y_i) \quad (2)$$

$$F2(x, y) = \begin{cases} 1 & if\ class(x) = class(y) \\ 0 & otherwise \end{cases} \quad (3)$$

In (1), ε denotes a threshold. For the experiment shown below, the value of ε=0.85 was used. This value was used since better results are obtained with it. The weights in expression (2) are calculated according to the method proposed in [24], and the features' comparison function $\partial_i(X_i, Y_i)$, which calculates the similarity between the values of objects *x* and *y* with respect to the feature

$$\partial_i (x_i, y_i) = \begin{cases} 1 - \dfrac{|(x_i - y_i)|}{Max(D_i) - Min(D_i)} & if\ i\ is\ continuous \\ 1 & if\ i\ is\ discrete\ and\ x_i = y_i \\ 0 & if\ i\ is\ discrete\ and\ x_i \neq y_i \end{cases} \quad (4)$$

Table 1. Description of data sets

Data sets	Amount of features	Amount of objects	Types of data
Breast-w	9	683	Continuous
Cleveland	13	297	Continuous
Ecoli	7	336	Continuous
Iris	4	150	Continuous
Heart-statlog	13	270	Continuous
Pima	8	768	Continuous
Wine	13	178	Continuous
Biomed	8	194	Continuous y discrete
Wisconsin	9	699	Continuous
Diabetes	8	768	Continuous
Pendigits	16	10992	Continuous
Wave form	40	5000	Continuous
Analcaa_bankruptcy	6	50	Continuous
Sonar	60	208	Continuous
Mfeat-zernike	47	2000	Continuous
Mfeat-fourier	76	2000	Continuous
Optdigits	64	5620	Continuous
Vehicle	16	846	Continuous
Cars	7	393	Continuous y discrete
Mfeat-factor	216	2000	Continuous

instances *i*, is defined by expression (4), where D_i is the domain of feature *i*.

Expressions (2) and (4) allow working with mixed data, i.e., application domains where the domain of descriptive features can include numeric and symbolic values.

For the experiments, 20 data sets taken from the UCI repository [25] were selected and partitioned using 10-fold cross validation.

In Table 1 the properties of the selected data sets are summarized.

The algorithm NP-BASIR-Class was compared with 12 algorithms mentioned in the article *A Taxonomy and Experimental Study on Prototype Generation for Nearest Neighbor Classification* [6]. They are the algorithms that offer best results according to [6] and are implemented in the tool KEEL [26]: LVQ3, GENN, DSM, VQ, MSE, ENPC, AVQ, PSCSA, Chen, HYB, PSO, and SGP. The results obtained show that the performance of NP-BASIR-Class gets good results in most cases and this is also supported by the comparison tests.

Table 2. Comparison of precision results of the obtained classification, for the methods of prototype generation

Data sets	NP-BASIR-Class	LVQ3	GENN	DSM	VQ	MSE	ENPC	AVQ	PSCSA	Chen	HYB	PSO	SGP
Breast-w(9)	96.95	94.87	96.20	95.31	95.61	96.93	95.03	96.34	96.93	93.86	94.59	96.49	89.17
Cleveland(13)	59.24	53.85	54.57	50.18	50.17	52.55	54.59	47.47	49.48	55.90	50.84	56.25	43.77
Ecoli(7)	81.87	68.51	81.60	64.88	62.50	78.61	74.40	72.98	62.58	71.11	74.10	80.69	64.30
Iris(4)	96.42	95.33	96.00	94.00	92.00	93.33	96.00	95.33	96.00	93.33	94.67	94.00	95.33
Heart-statlog(13)	81.48	80.37	77.04	70.74	70.74	78.89	76.67	82.96	78.15	78.89	70.74	82.22	61.11
Pima(8)	75.38	73.96	72.91	65.88	67.83	72.93	64.70	70.82	75.65	70.57	65.09	75.38	59.64
Wine(13)	96.81	94.97	96.11	92.12	91.54	94.41	96.67	94.97	90.39	95.52	95.52	97.22	93.86
Biomed(8)	95.98	87.13	94.34	91.32	87.16	92.29	92.32	89.18	84.53	89.76	94.32	87.18	85.42
Wisconsin(9)	96.83	95.32	96.35	95.02	94.43	96.93	94.59	96.05	97.08	93.85	95.32	96.34	90.48
Diabetes(8)	76.69	73.70	72.52	66.93	66.67	74.09	63.54	70.19	73.97	68.10	63.54	75.78	62.24
Pendigits(16)	95.88	94.26	99.35	97.82	97.82	97.07	99.02	89.27	68.34	98.92	97.95	96.21	91.69
Wave form (40)	83.78	78.84	76.16	70.64	70.26	82.52	78.04	84.22	66.40	74.38	77.22	83.94	71.94
Analcaa_bankruptcy(6)	92.00	–	84.00	86.00	62.00	76.00	78.00	76.00	78.00	84.00	82.00	–	72.00
Sonar(60)	82.54	66.24	85.95	74.00	65.83	76.86	88.90	72.48	65.76	64.38	84.10	73.55	71.10
Mfeat-zernike(47)	81.05	73.75	79.90	75.30	74.20	80.55	76.35	74.95	46.95	72.95	77.35	79.42	73.70
Mfeat-fourier(76)	81.50	73.05	81.40	70.90	70.35	80.20	79.80	76.15	43.90	67.15	79.30	67.75	76.55
Optdigits(64)	97.47	94.84	98.67	96.46	96.51	97.56	98.13	93.36	62.81	95.34	98.13	93.86	93.47
Vehicle(16)	68.72	54.48	68.68	59.32	58.75	65.72	66.90	53.54	43.62	58.28	70.56	64.79	55.79
Cars(7)	77.83	66.27	74.99	71.38	63.99	73.94	72.92	63.51	65.76	68.09	73.97	72.67	57.17
Mfeat-factor(216)	96	89.70	95.85	91.15	91.10	95	95.20	90.40	68.05	89.35	95.80	91.9	85.45

To calculate the measurement accuracy we used the measure that is typically used in these cases as shown in expression (5):

$$Accuracy = \frac{(TP + TN)}{(TP + TN + FP + FN)} \qquad (5)$$

where TP are True Positives, TN are True Negatives, FP are False Positives, and FN are False Negatives.

To compare the results, multiple comparison tests were used in order to find the best algorithm using all data sets.

The reduction coefficient Re (·) [15] was studied also. This measure, see expression (6), indicates in what quantity the number of the objects is reduced (relation between the size of the set of prototypes (P) and the amount of instances(X)).

Table 2 shows, for each method, the accuracy of the classification of the 12 selected methods and the one proposed in this paper, NP-BASIR-Class, after applying them on the 20 data sets described above, where it can be seen that in most cases the proposed method achieves better results. The "–" symbol denotes that for this data set the algorithm did not obtain any result because it exceeded the computational time threshold.

5 Discussion

In this section we analyze the results obtained, specifically, we check the performance of the NP-BASIR-Class method and the twelve other PG techniques.

Table 3 shows that the algorithm NP-BASIR-Class obtains the best ranking. The Iman-

Table 3. Ranking obtained by the Friedman's Test

Algorithms	Ranking
NP-BASIR-Class	2.225
GENN	3.675
MSE	5.1
ENPC	5.575
PSO	5.625
HYB	6.05
AVQ	7.8
LVQ3	8.225
DSM	8.05
Chen	8.5
PSCSA	9.35
VQ	9.875
SGP	10.95

Table 4. Holm's table for α = 0.05, with NP-BASIR-Class as control method

i	Algorithms	z=(R0−Ri)/SE	p	Holm	Hypothesis
12	SGP	7.084682	0	0.004167	Rejected
11	VQ	6.211784	0	0.004545	Rejected
10	PSCSA	5.785485	0	0.005	Rejected
9	Chen	5.095287	0	0.005556	Rejected
8	LVQ3	4.871988	0.000001	0.00625	Rejected
7	DSM	4.729888	0.000002	0.007143	Rejected
6	AVQ	4.526889	0.000006	0.008333	Rejected
5	HYB	3.105892	0.001897	0.01	Rejected
4	PSO	2.760793	0.005766	0.0125	Rejected
3	ENPC	2.720193	0.006524	0.016667	Rejected
2	MSE	2.334494	0.01957	0.025	Rejected
1	GENN	1.177397	0.239037	0.05	Rejected

Davenport's Test (*F*-distribution with 13 and 234 degrees of freedom) was employed in order to find significant differences among the algorithms, obtaining by the Friedman's Test a *p*-value of 0. Thus, in Table 4 the results of NP-BASIR-Class are compared with other methods using the Holm's Test [27]. The test rejects all cases for better ranking algorithm, so it can be seen that the proposed algorithm is statistically higher than the others in terms of classification accuracy.

In Table 5, the reduction coefficient obtained for each method is shown for the large data sets (more than 1000 instances). The second column (called Red) presents the reached reduction. In this case our algorithm obtains 90% in the reduction coefficient that we can consider as a good result.

Table 6 shows the average of time (in seconds) obtained for the methods of prototype generation over the large data sets (Pendigits, Waveform, Mfeat-zernike, Mfeat-fourier, Optdigits, Mfeat-

Table 5. Average of obtained results for the methods of prototype generation over the large data sets (Pendigits, Waveform, Mfeat-zernike, Mfeat-fourier, Optdigits, Mfeat-factor)

Methods of prototype generation	Red
GENN	15.76
HYB	57.27
ENPC	82.85
NP-BASIR-Class	90.56
LVQ3	97.99
DSM	97.99
VQ	97.99
PSO	97.99
Chen	98.01
SGP	98.23
MSE	99.36
AVQ	99.8
PSCSA	99.88

Table 6. Average of obtained time for the methods of prototype generation over the large data sets (Pendigits, Waveform, Mfeat-zernike, Mfeat-fourier, Optdigits, Mfeat-factor)

Methods of prototype generation	Time (s)
LVQ3	10.99
DSM	36.53
SGP	42.5
NP-BASIR-Class	80.90
VQ	99.17
AVQ	480
HYB	620.19
Chen	658.96
MSE	1020
GENN	1132.32
ENPC	1560.16
PSCSA	2450.01
PSO	3980.75

factor). The proposed algorithm obtains favorable results in the time of execution with regard to the other algorithms.

In Table 7, the amount of prototypes generated by the algorithm NP-BASIR-Class and the reduction coefficient obtained for each data set are presented. The number of prototypes was calculated based on the average of the quantity of prototypes generated for each partition (10 partitions). Note that the best results in the

Table 7. Amount of generated prototypes and the reduction coefficient for each data set

Data sets	Number of objects	Number of prototypes	Reduction coefficient
Breast-w	683	58.7	91.40
Cleveland	297	96.8	67.40
Ecoli	336	74.9	77.70
Iris	150	7	95.33
Heart-statlog	270	62.4	76.88
Pima	768	146.3	80.95
Wine	178	36.4	79.55
Biomed	194	31.4	83.81
Wisconsin	699	26.5	96.20
Diabetes	768	165.4	78.46
Pendigits	10992	355.8	96.76
Wave form	5000	41.7	99.16
Analcaa_bankruptcy	50	3.6	92.80
Sonar	208	34.3	83.51
Mfeat-zernike	2000	355.9	82.20
Mfeat-fourier	2000	355.9	82.20
Optdigits	5620	422.6	92.48
Vehicle	846	183.2	78.34
Cars	393	76	80.66
Mfeat-factor	2000	330.6	83.47

reduction coefficient were for the bases with more objects: Waveform with 5,000 objects (99.16%) and Pendigits with 10,992 objects (96.76%). In this table the results show that the reduction coefficient obtained by the proposed algorithm NPBASIR-Class is significant.

6 Conclusions

In this paper a new classification method based on nearest prototypes (NP-BASIR-Class) is presented. which includes an algorithm for constructing prototypes using similarity classes. To evaluate the results of this method. statistical tests for multiple comparisons were used in order to find the best algorithm.

Experimental results show that the proposed method has the best ranking. and it is statistically higher than the others in terms of classification accuracy. The reduction factor and the time of execution achieved were also analyzed showing satisfactory results.

References

1. **García-Duran, R., Fernández, F., & Borrajo, D. (2012).** A prototype-based method for classification

with time constraints: a case study on automated planning. *Pattern Analysis & Applications,* Vol. 15. No. 3, pp. 261–277, doi: 10.1007/s10044-010-0194-6

2. **Barandela, R., Cortes, N., & Palacios, A. (2001).** The nearest neighbor rule and the reduction of the training sample size. *Proceedings 9th Symposium on Pattern Recognition and Image Analysis.* Castellon. Spain, Vol. 1, pp. 103–108.
3. **James, C., Bezdek, J., & Kuncheva, L. (2001).** Nearest Prototype Classifier Designs: An Experimental Study. *International Journal of Intelligent Systems*, Vol. 16, pp.1445–1473, doi: 10.1002/int.1068
4. **Kim, S.W. & Oommen, B.J. (2003).** A brief taxonomy and ranking of creative prototype reduction schemes. *Pattern Anal Applications,* Vol. 6, pp. 232–244.
5. **García, S., Cano, J.R., & Herrera, F., (2008).** A memetic algorithm for evolutionary prototype selection: A scaling up approach. *Pattern Recognition*, Vol. 41. No. 8, pp. 2693–2709. doi: 10.1016/j.patcog.2008.02.006
6. **Triguero, I., Derrac, J., García, S., & Herrera, F. (2012).** A Taxonomy and Experimental Study on Prototype Generation for Nearest Neighbor Classification. *IEEE Transactions on Systems. Man. and Cybernetics--Part C: Applications and Reviews*, Vol. 42, No. 1, pp. 86–100.
7. **Wilson, D.R. & Martínez, T.R. (2000).** Reduction tecniques for instance-based learning algorithms. *Machine Learning,* Vol. 38, pp. 257–268, doi: 10.1023/A:1007626913721
8. **Liu, H. & Motoda, H. (2002).** On issues of instance selection. *Data Mining and Knowledge Discovery*, Vol. 6, pp. 115–130.
9. **Olvera, J.A., Carrasco, J.A., & Martínez, J.F. (2010).** Prototype Selection Methods. *Computación y Sistemas,* Vol. 13, No. 4, pp. 449–462.
10. **Triguero, I., Derrac, J., García, S., & Herrera, F. (2012).** Integrating a Differential Evolution Feature Weighting scheme into Prototype Generation. *Neurocomputing,* Vol. 97, pp. 332–343, doi: 10.1016/j.neucom.2012.06.009
11. **Bello-Garcia, M., Garcia, M.M., & Bello, R. (2013).** A Method for Building Prototypes in the Nearest Prototype Approach Based on Similarity Relations for Problems of Function Approximation. *LNAI,* Vol. 7629, Springer-Verlag, pp. 39–50.
12. **Martínez-Trinidad, J. Francisco, & Yao, Y.Y. (2000).** Granular computing: basic issues and possible solutions. *Proceedings of the 5th Joint Conference on Information Sciences,* pp. 186–189.
13. **Filiberto, Y., Bello, R., Caballero, Y., & Larrua, R. (2010).** A method to built similarity relations into extended Rough set theory. *Proceedings of the 10th International Conference on Intelligent Systems Design and Applications (ISDA2010),* pp. 1314–1319.
14. **Duda, R.O. & Hart, P.E. (1973).** *Pattern Classification and Scene Analysis*. John Wiley and Sons.
15. **Bermejo, S. & Cabestany, J**. **(2000).** A Batch Learning Algorithm Vector Quantization Algorithm for Nearest Neighbour Classification. *Neural Processing Letters,* Vol. 11, pp. 173–184, doi: 10.1023/A:1009634824627
16. **Patané, G. & Russo, M**. **(2001).** The Enhanced LBG Algorithm. *Neural Networks*, Vol. 14, pp. 1219–1237, doi:10.1016/S0893-6080(01)00104-6
17. **Kuncheva, L.I. & Bezdek, J.C**. **(1998).** Nearest Prototype Classification: Clustering. Genetic Algorithms, or Random Search*? IEEE Transactions on Systems. Man and Cybernetics*, Vol. 28, No. 1, pp. 160–164, doi: 10.1109/5326.661099
18. **Kuncheva, L.I. & Bezdek, J.C. (1998).** An integrated framework for generalized Nearest prototype classifier design. *International Journal of Uncertainty. Fuzziness and Knowledge Based Systems*, Vol. 6, No. 5, pp. 437–457, doi: 10.1142/S0218488598000355
19. **Fernández, F. & Isasi, P. (2004).** Evolutionary Design of Nearest Prototype Classifiers. *Journal of Heuristics*, Vol. 10, pp. 431–454, doi: 10.1023/B:HEUR.0000034715.70386.5b
20. **Kohonen, T. (1989).** *Self-Organization and Associative Memory*. Springer.
21. **Fritzke, B. (1994).** Growing Cell Structures—A Self-Organizing Network for Unsupervised and Supervised Learning. *Neural Networks,* Vol. 7, No. 9, pp. 1441–1460.
22. **Olvera-López, J. & Carrasco-Ochoa, J.A. (2010).** A new fast prototype selection method based on clustering. *Pattern Anal Applic*., Vol. 13, pp. 31–141, doi: 10.1007/s10044-008-0142-x
23. **López, R.L. & Armengol, E. (1998).** Machine learning from examples: Inductive and Lazy methods. *Data & Knowlege Engineering*, Vol. 25, pp. 99–123, doi: 10.1016/S0169-023X(97)00053-0
24. **Filiberto, Y. & Bello, R. (2010).** Using PSO and RST to Predict the Resistant Capacity of Connections in Composite Structures. *International Workshop on Nature Inspired Cooperative Strategies for Optimization* (*NICSO 2010*)*,* Springer, pp. 359–370.

25. **Blake, C.L. & Merz, C.J. (1998).** *UCI repository of machine learning databases*. http://www.ics.uci.edu/~mlearn/MLRepository.html
26. **Alcalá-Fdez, J., Sánchez, L., García, S. del Jesús. M.J., Ventura, S., Garrell, J., Otero, J., Romero, C. Bacardit, J., Rivas, V.M., Fernández, J.C., & Herrera, F. (2008).** KEEL: A software tool to assess evolutionary algorithms for data mining problems. *Soft Comput.*, Vol. 13, No. 3, pp. 307–318.
27. **Holm, S. (1979).** A simple sequentially rejective multiple test procedure. *Journal of Statistics*, Vol. 6, pp. 65–70.

Yumilka B. Fernández Hernández received her Bachelor degree in Computer Science in 2004 from the Universidad de Camagüey(UC),Cuba, and MSc. in Telecommunication in 2006 from the Universidad Central de Las Villas (UCLV), Cuba. Her scientific interest is Artificial Intelligence, in particular, machine learning, soft computing, and decision making. She participated in several international congresses. She is a member of the Research Group on Artificial Intelligence

Rafael Bello received his Bachelor degree in Cybernetic and Mathematics from the Universidad Central de Las Villas (UCLV), Cuba, in 1982, and his PhD degree in 1987. His scientific interest is Artificial Intelligence, in particular, metaheuristics, soft computing, machine learning, and decision making. He published more than 200 scientific papers. He is Member of the Cuban Academy of Sciences and Director of the Center of Studies on Informatics of the UCLV.

Yaima Filiberto received her Bachelor degree in Computer Science in 2006 and> > MSc. in Applied Computer Science in 2008 both from the Universidad de Camagüey (UC), Cuba, and her PhD degree in 2012 from the Universidad Central de Las Villas (UCLV) Cuba. Her scientific interest is Artificial Intelligence, in particular, machine learning, soft computing, KDD and decision making. She published more than 30 scientific papers. She is Scientific and Technical Director of the UC.

Mabel Frías received her Bachelor degree in Computer Science from the Universidad de Camaguey, Cuba, in 2011. Later she received the category of master in teaching mathematics in 2014. She is professor of the Faculty of Computer Science. Her scientific interest is Artificial intelligence, in particular, machine learning. She is a member of the Research Group on Artificial Intelligence.

Lenniet Coello Blanco received her Bachelor degree in Computer Science from the Universidad de Camaguey, Cuba, in 2011. Later she received the category of master in teaching mathematics in 2014. She is professor of the Faculty of Computer Science. She participated in 15 congresses, most of them international. Her scientific interest is Artificial Intelligence, in particular, machine learning and experts systems. She is a member of the Research Group on Artificial Intelligence.

Yaile Caballero received her Bachelor degree in Computer Science from the Universidad Central de Las Villas (UCLV), Cuba, in 2001, and her PhD degree in 2007. Her scientific interest is Artificial Intelligence, in particular, metaheuristics, soft computing, machine learning, and decision making. She published more than 140 scientific papers. She is Member of the Cuban Academy of Sciences.

Article received on 02/10/2014, accepted on 20/01/2015. Corresponding author is Yumilka B. Fernández.